ELSEVIER

Mathematics and Computers in Simulation 57 (2001) 307–315

MATHEMATICS AND COMPUTERS IN SIMULATION

www.elsevier.com/locate/matcom

Communications between the Poisson series processor PSPC and general scientific software

J.F. San Juan [a,b,*], Alberto Abad [a]

[a] *Grupo de Mecánica Espacial, Univ. de Zaragoza, 50009 Zaragoza, Spain*
[b] *Dept. de Matemáticas y Computación, Univ. de la Rioja, 26004 Logroñno, Spain*

Abstract

Special efficient Poisson series processors (PSP) have been created. Poison series appear frequently in problems of non-linear dynamics and celestial mechanics and their size makes their manipulation by means of general computer algebra systems (CAS) inefficient, but the characteristics of the problems suggest the use of general CAS with other general scientific software like compilers, libraries of compiled programs or word processors. To combine these specialized and general tools we need to communicate between them. In this paper we describe the tools to share information between PSPC, our own PSP, and other general software. © 2001 IMACS. Published by Elsevier Science B.V. All rights reserved.

Keywords: Computer algebra systems; Poisson series processors

1. Introduction

In a previous paper [2,4,7] we described ATESAT, a software system to obtain automatically analytical theories in the artificial satellite problem. The core of ATESAT is a specialized computer algebra system, named PSPC, used to handle Poisson series that appear frequently in non-linear mechanics, and in particular, in celestial mechanics. Let us recall that a Poisson series is a multivariate Fourier series of the form

$$\sum_{i_0,\ldots,i_{n-1},j_0,\ldots,j_{m-1}} C_{i_0,\ldots,i_{n-1}}^{j_0,\ldots,j_{m-1}} x_0^{i_0}\cdots x_{n-1}^{i_{n-1}} \begin{pmatrix}\sin\\ \cos\end{pmatrix}(j_0y_0+\cdots+j_{m-1}y_{m-1}).$$

This series consists of a rational or real coefficient, n polynomial variables with integer exponents and a trigonometric function of a linear combination of angular variables. The set $\mathcal{P}$ of Poisson series is a commutative algebra with a unit element. We call it series by an abuse of language, because although it

* Corresponding author.
E-mail addresses: juanfelix.sanjuan@dmc.unirioja.es (J.F. San Juan), abad@posta.unizar.es (A. Abad).

0378-4754/01/$ – see front matter
PII: S0378-4754(01)00345-7

may have a finite or infinite number of terms, in practice, in order to define an equivalent computational object, we need to consider only series with a finite number of terms.

Usually, high-order theories in celestial mechanics include series with a huge number of terms. For instance, in ATESAT around 1,200,000 algebraic terms are manipulated when the three Lie transformations (elimination of the parallax, elimination of the perigee and Delaunay normalization) are applied to the zonal problem of the artificial satellite in order to create a third-order theory.

In PSPC the algebraic structure of Poisson series is created, and the operations among Poisson series is extended to the differential and integral calculus and other specialized operations such as substitutions of variables by series, splitting series depending on certain conditions, etc. A more detailed description of PSPC can be found in [1,5].

Once PSPC is applied for solving a problem, the result obtained is one or more Poisson series with variable size depending on the problem. Usually these series are used in

- the qualitative analysis of the results by using other computer algebraic systems, like *Mathematica* for instance, with more general tools included graphics;
- the evaluation of a theory by making a quantitative study of the problem by means of a compiled program, using languages like C or FORTRAN;
- a scientific word processor, like LATEX, to have a comprehensive print copy with the results of the theory.

Thus, manipulation of the results is needed to write these series in a format compatible with a software different than PSPC. An automatic translation of the series would avoid the possible errors of a manual manipulation. We describe bellow the characteristics of the translation of the Poisson series generated by PSPC to a format readable by *Mathematica*, C and LATEX.

In the case of LATEX, the problem is a simple translation between two different languages. However, some other aspects more involved than the simple translation must be taken into account when the Poisson series obtained by PSPC are used in a C program, or in a *Mathematica* session.

Considering the large size of the series appearing in particular problems, like the ones handled by ATESAT, we need a way to write automatically an efficient C or FORTRAN code to evaluate them. The efficiency will consist of reducing as much as possible the number of calls to the mathematical library, minimizing the evaluation of powers and trigonometric functions, and defining good data structures. An algorithm of Coffey and Deprit [6] combined with an internal storage format of the series in PSPC, different from the ones used in other PSPs, was applied to make a more efficient code.

PSPC can collaborate with *Mathematica* [9] in two different ways. The simplest one is the translation of the series to the *Mathematica* format, in order to be used in a *Mathematica* session as a usual *Mathematica* object. A more involved way is to build, by means of the communication protocol MathLink [8], a new kernel of *Mathematica* that we named PSPCLink [3] such that the Mathematica front-end send all the operations involving Poisson series to PSPCLink, whereas the rest of operations are sent to the kernel of *Mathematica*. With this tool we combine the flexibility of *Mathematica* with the efficiency of PSPC.

2. Symbolic representation of a Poisson series

Let us give a more precise definition of Poisson series.

Definition 1. We call *Poisson series* of n polynomial variables $\boldsymbol{x} = x_1, \ldots, x_n$ and m angular variables $\boldsymbol{y} = y_1, \ldots, y_m$, to a map $(\boldsymbol{x}, \boldsymbol{y}) \to \mathcal{S}(\boldsymbol{x}, \boldsymbol{y}) : \boldsymbol{R}^n \times \boldsymbol{R}^m \to \boldsymbol{R}$, defined by

$$\mathcal{S}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\boldsymbol{i}\in\mathcal{I}, \boldsymbol{j}\in\mathcal{J}} C_{\boldsymbol{i}}^{\boldsymbol{j}} P_{\boldsymbol{i}} T_{\boldsymbol{j}}, \quad C_{\boldsymbol{i}}^{\boldsymbol{j}} \in \boldsymbol{R},$$

where $\boldsymbol{i} = i_0, \ldots, i_{n-1}$ and $\boldsymbol{j} = j_0, \ldots, j_{m-1}$ are elements belonging to $\boldsymbol{Z}^n$ and $\boldsymbol{Z}^m$, respectively, and furthermore

$$P_{\boldsymbol{i}} = x_0^{i_0} \cdots x_{n-1}^{i_{n-1}} \quad \text{and} \quad T_{\boldsymbol{j}} = \begin{pmatrix} \sin \\ \cos \end{pmatrix} (j_0 y_0 + \cdots + j_{m-1} y_{m-1}),$$

represent the polynomial and trigonometric parts, respectively.

Each term of a Poisson series is characterized by the following basic information:

1. A number C_i^j that represents the coefficients.
2. n integers $(i_0, \ldots, i_{n-1})$ that represent the exponents of the polynomial part P_i and characterize it.
3. m integers $(j_0, \ldots, j_{m-1})$ that represent the angular variable coefficients of the trigonometrical part T_j, plus an integer, or simply a bit, equal to 0 or 1 which says whether the term is a sine or a cosine. The $m + 1$ integers characterize T_j.

The symbolic representation of these mathematical objects is closely connected with the computational data structure. It determines the efficiency of the PSP in handling series, as well as the efficiency of the C code generated to evaluate them.

The classical symbolic representation

$$\sum_{\boldsymbol{i}\in\mathcal{I}\boldsymbol{j}\in\mathcal{J}} C_{\boldsymbol{i}}^{\boldsymbol{j}} P_{\boldsymbol{i}} T_{\boldsymbol{j}} \tag{1}$$

is based on the lexicographical ordering of the terms. This ordering is similar to the one obtained by *Mathematica* when the expression `Expand[TrigReduce[_]]` is applied.

Fig. 1 shows the list structure usually used to store Poisson series. This data structure corresponds naturally to the classical symbolic representation.

Another symbolic representation of these series is obtained when the terms with a common trigonometric part are grouped together

$$S(\boldsymbol{x}, \boldsymbol{y}) = \sum_{j=1}^{\mathrm{card}(\mathcal{J})} \left(\sum_{i=1}^{\mathrm{card}(\mathcal{I})} C_i^j P_i \right) T_j. \tag{2}$$

This expression can be written in a matrix form as

$$S(\boldsymbol{x}, \boldsymbol{y}) = PCT, \tag{3}$$

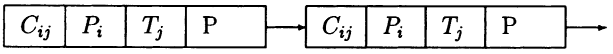

Fig. 1. Unidimensional list structure.

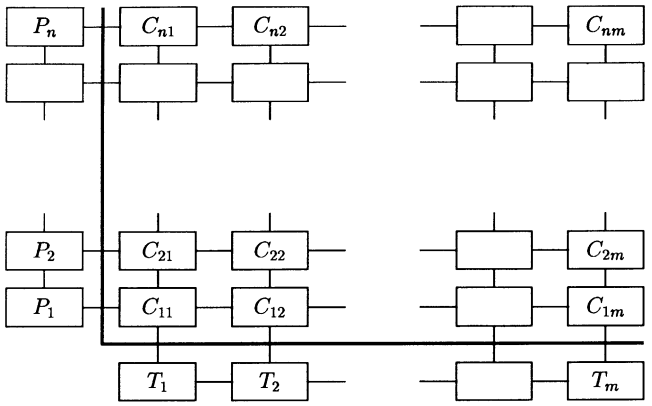

Fig. 2. Bidimensional list structure used in PSPC.

where $P = \{P_i, i = 1, \ldots, \mathrm{card}(\mathcal{I})\}$, $T = \{T_j, j = 1, \ldots, \mathrm{card}(\mathcal{J})\}$, and C is the matrix $(C_{i,j})$. This expression (3) represents a new useful symbolic representation of the series.

In PSPC, the data structure implemented belongs to the matrix representation. This structure is shown in Fig. 2. The use of this representation will improve the strategy in evaluating the series.

3. Evaluation of series

One on the goals of PSPC is the possibility of handling series with a large number of terms. For instance, in the artificial satellite theory developed by ATESAT for a zonal problem, the expression of the radial distance after the elimination of the perigee, contains more than 80,000 terms, and the expression of the argument of latitude is a series with more than 150,000 terms. Using such data in a quantitative study, creates a non trivial problem.

Two aspects of the problem must be considered. On the one hand is the automatic construction of the code (FORTRAN or C) to evaluate an expression. On the other is the design of the internal algorithm for evaluating the series, which is crucial when the series are large. The automatic construction of the code has been traditionally considered in general mathematical software. *Mathematica*, for instance, have the commands `CForm` and `FortranForm` to transform a *Mathematica* expression into a C or FORTRAN code. The main contribution to the problem of the evaluation, given by specialized computer algebra systems like PSPC, is the possibility of taking into account the inherent properties of the object in order to build a faster algorithm to evaluate large series.

To illustrate the historical approach to the problem of the evaluation of Poisson series, let us consider the series resulting from the following operation: $S_1^2 + S_2^2$, where $S_1 = (x + y)\sin(a - b)$, and $S_2 = (x - y)\cos(a + b)$. Using the *Mathematica* order `Expand[TrigReduce[_]]` we obtain the symbolic expression

$$x^2 + y^2 - \tfrac{1}{2}x^2\cos(2a - 2b) - xy\cos(2a - 2b) - \tfrac{1}{2}y^2\cos(2a - 2b) + \tfrac{1}{2}x^2\cos(2a + 2b) - xy\cos(2a + 2b) + \tfrac{1}{2}y^2\cos(2a + 2b) \quad (4)$$

corresponding to the classical symbolic representation (1) of the Poisson series. The function `FortranForm` gives us the FORTRAN code to evaluate a series like

```
x**2+y**2-(x**2*COS(2*a-2*b))/2.-x*y*COS(2*a-2*b)
      -(y**2*COS(2*a-2*b))/2.+(x**2*COS(2*a+2*b))/2.
      -x*y*COS(2*a+2*b)+(y**2*COS(2*a+2*b))/2.
```

This FORTRAN code corresponds to the first, not optimized, approach in the evaluation of series by the Poisson series processors. The list structure in which the series are stored does not permit to distinguish common trigonometric factors between the different terms, and the expressions $\cos(2a + 2b)$ and $\cos(2a - 2b)$ are evaluated repeatedly. The excessive use of the library functions `SIN` and `COS` makes very inefficient the generated code to evaluate large series.

The second approach to the evaluation of series maintains the list structure, but minimizes the number of calls to the routines `SIN` and `COS` by applying an algorithm developed by Coffey and Deprit [6]. This algorithm, named *navigation table*, makes use of the addition theorems of the trigonometric functions to create a table for computing the trigonometric terms by means of sums and products and only one call to the `SIN` and `COS` functions for each variable.

To illustrate the last approach to the evaluation of series let us apply to the previous idea the *Mathematica* expression

```
Collect[Expand[TrigReduce[\_]], \{Sin[\_], Cos[\_]\}].
```

We obtain, instead of Eq. (4), the expression

$$(x^2 + y^2) - \left(\tfrac{1}{2}x^2 + xy + \tfrac{1}{2}y^2\right)\cos(2a - 2b) + \left(\tfrac{1}{2}x^2 - xy + \tfrac{1}{2}y^2\right)\cos(2a + 2b)$$

whose FORTRAN code is

```
x**2+y**2+(-x**2/2.-x*y-y**2/2.)*COS(2*a-2*b)
      +(x**2/2.-x*y+y**2/2.)*COS(2*a+2*b)
```

This last expression corresponds to the symbolic representation of the Poisson series given in Eq. (2). This representation avoids the repetition of trigonometric terms in both the internal storage of series, using the bidimensional structure (Fig. 2), and the evaluation of series.

In spite of that, we use the navigation table combined with the bidimensional representation to maximize as much as possible the efficiency of the code. The power of the polynomial variables is computed by products and stored in arrays of powers. Finally, we use the matrix product to compute the result. We use either a sparse or a dense matrix depending on the ratio between zeros and non-zeros in the matrix.

As a real example, let us see for instance the series $H_{1,2}$ appearing in ATESAT when we apply the elimination of the parallax to the satellite problem. This is a small series with 156 polynomial terms, 19 trigonometric terms and 460 coefficients (2964 if we include zeros). The classical list representation requires 460 evaluations of the trigonometric functions, while the bidimensional representation implemented in PSPC requires only 19 evaluations of the trigonometric functions. This combined with the navigation table, also implemented in PSPC, and a sparse representation of the matrix of coefficients, improves notably the efficiency in the evaluation of Poisson series.

4. PSPCLink

The specialized computer algebra systems like PSPC, has been developed to take advantage of the particular properties of the mathematical objects they manipulate. This is particularly important when the object can have a large size as in the case of Poisson series. General tools like *Mathematica* do not consider these particular properties, and they loose efficiency in size of storage and time of computation with respect to the specialized tools. In particular, there are a lot of cases handled with ATESAT that are impossible to be solved with *Mathematica*.

However, *Mathematica* and other mathematical software tools provide a lot of important possibilities like a friendly front-end, the possibility of handling simultaneously a great variety of mathematical objects, including special functions, graphics, etc. This is very useful in the early stages of the development of an analytical theory, and also to analyze the final results of these theories.

To combine the possibilities of both tools, the first solution is the definition of a common format to interchange Poisson series between *Mathematica* and PSPC. However, *Mathematica* provides the possibility of using a C compiled code by means of the communication protocol *MathLink*. This protocol will give us the possibility to use PSPC inside *Mathematica*.

The front-end of *Mathematica* is the interface between the user and *Mathematica*. One writes an expression in the front-end and it sends the expression to the kernel, a standalone application, where this expression is evaluated sending the final result to the front-end again. By using the communications protocol *MathLink* we wrote PSPCLink. PSPCLink is an application that contains PSPC and that acts as a second kernel of *Mathematica*.

As we can see in Fig. 3, when we write an expression in the front-end that contains a Poisson series, it sends the expression to PSPCLink instead of the standard kernel.

To create PSPCLink we need all the PSPC code to be included in the file `PSPC.c`; *MathLink* does not permit user-defined types, for this reason we need to create a new file `PSPCLink.c` to synchronize the user data types with the data types recognized by *MathLink*. Finally, a file `PSPCLink.tm` that contains the functions in *Mathematica* and the names of the corresponding functions in C and the types of the arguments, is preprocessed with `mprep`, a part of *MathLink*, and we obtain a file `PSPCLink.tm.c` that is compiled together with `PSPC.c` and `PSPCLink.c` to create the application PSPCLink (Fig. 4).

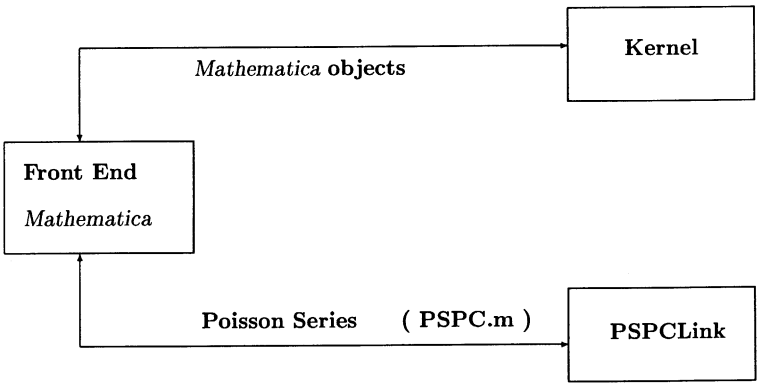

Fig. 3. PSPCLink as a second kernel of *Mathematica*.

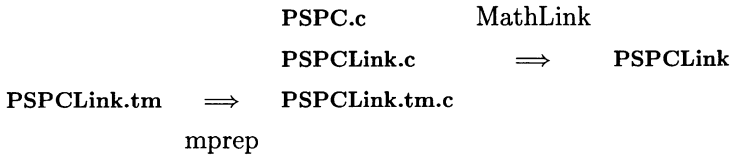

Fig. 4. Scheme of generation of PSPCLink.

As an example, we can see how to add two Poisson series. The first step is to have, inside `PSPC.c`, a function `add_series` to add the given series, by means of three pointers, to a PSPC data type `SERIES` as argument.

```
int add_series(a, b, c)
SERIES * a, *b, *c
{
  ...
}
```

MathLink cannot pass these kind of arguments from the front-end to PSPCLink. For this reason we wrote the function `AddSeries` included in the file `PSPCLink.c`. In this function an integer number is used to identify each series. By means of these integer numbers we obtain a pointer to the series and we call, inside it, the function `add_series`.

```
int AddSeries (n, m, s)
int n, m, s
{
  return(add_series(pointers_to → n, m, s))
}
```

In `PSPCLink.tm` we declare the names of the function directly connected by *MathLink*.

```
:Begin:
:Function: AddSeries
:Pattern: AddSeries[n_Integer, m_Integer, s_Integer]
:Arguments: {n, m, s}
:ArgumentTypes: {Integer, Integer, Integer}
:ReturnType: Integer
:End :
```

Once we have installed PSPCLink in *Mathematica*, we pass the Poisson series from the front-end to the new kernel by using the expressions `ToPoissonSeries` and `FromPoissonSeries`. For instance,

```
ps1 = ToPoissonSeries[1 + x Sin[a − b]]
```

sends the series $1 + x \sin(a - b)$ to PSPCLink to store it, and returns to the front-end the expression PoissonSeries[1], whose head identifies the object as a Poisson series, and the integer argument 1 is used to access the stored series. If we have another series `ps2` with the index 2 we may add both series writing

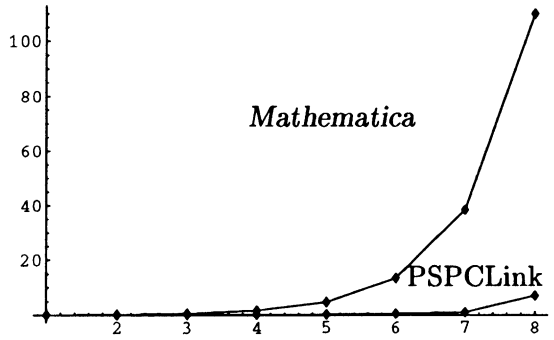

Fig. 5. Computational time (s) of the expression (5) with PSPCLink and only with *Mathematica*, changing the value of n from 1 to 8.

```
AddSeries[1, 2, 3]
```

that stores the addition in a Poisson series identified with the index 3.

Instead of using the expression `AddSeries` to add series, like in procedural languages, we prefer to use the *Mathematica* style. With this style we add the series with the operator +. In order to do that, we overload this operator by changing the properties of the head `Plus` associated with the symbol `PoissonSeries`.

To show the efficiency of PSPCLink versus the exclusive use of the *Mathematica* functions, we present a graph (Fig. 5) with the computational time of the following operation:

$$\left[1 + \left(\sum_{i=1}^{n} x_i\right) \cos(a + b)\right]^5 \tag{5}$$

in which we change the number n of polynomial variables.

The calculations have been made with a PowerPC 750 G3, 266 MHz, and 30MB of RAM memory dedicated to *Mathematica*. We see in the graphic that the efficiency in using PSPCLink increases as the number of variables does. The time of computation decreases notably when we made the same operation with a compiled standalone application created with PSPC since the communications between the front-end of *Mathematica* and PSPCLink is relatively slow, but with PSPCLink we have the possibility of handling Poisson series from *Mathematica* in a faster way than by using only *Mathematica*.

Acknowledgements

We are very grateful to J.C. Agnesse for reporting bugs, testing routines, etc. We also thank A. Elipe for his valuable suggestions, and I. Tijera for translating the manuscript. This work has been supported in part by the Ministerio de Educación y Ciencia (DGICYT Project PB95-0807), the University of La Rioja (Project API-98/A11) and the Department of Space Mathematics of Centre National d'Etudes Spatiales (France).

References

[1] A. Abad, J.F. San Juan, PSPC: a Poisson series processor coded in C, in: S. Kurzyńska, et al. (Eds.), Dynamics and Astrometry of Natural and Artificial Celestial Bodies, Poznan, Poland, 1993, pp. 383–389.
[2] A. Abad, J.F. San Juan, in: A. Elipe, P. Pâquet (Eds.), ATESAT: Software Tool for Obtaining Automatically Ephemeris from Analytical Simplifications, Vol. 10, Conseil de L'Europe, Cahiers du Centre Européen de Géodynamique et de Séismologie, Luxembourg, 1995, pp. 93–98.
[3] A. Abad, J.F. San Juan, PSPCLink: a cooperation between general symbolic and Poisson series processors, J. Symbolic Comput. 24 (1997) 113–122.
[4] A. Abad, A. Elipe, J. Palacián, J.F. San Juan, ATESAT: a symbolic processor for artificial satellite theory, Math. Comput. Simulat. 45 (1998) 497–510.
[5] A. Abad, J.F. San Juan, Algebraic and symbolic manipulation of Poisson series, J. Symbolic Comput. 1999, submitted for publication.
[6] S.L. Coffey, A. Deprit, Fast evaluation of Fourier series, Astron. Astrophys. 81 (1980) 310–315.
[7] J.F. San Juan, Manipulación algebraica de series de Poisson. Aplicación a la teoría del satélite artificial, Ph.D. thesis, Universidad de Zaragoza, 1996.
[8] Wolfram Research, MathLink Reference Guide, Technical Report, 1991.
[9] S. Wolfram, The *Mathematica* Book, Wolfram Media Inc., 1996.